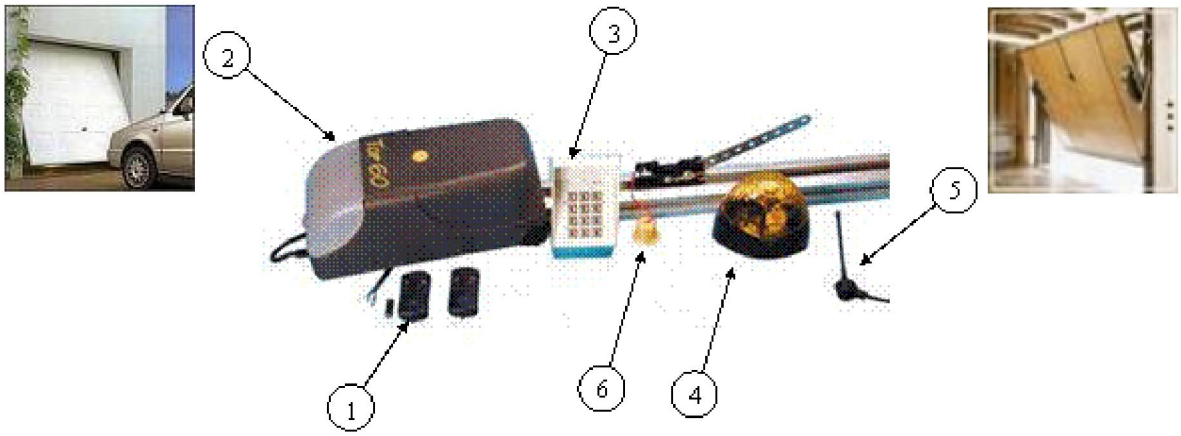
Leçon A4-2 : Les Microcontrôleurs

Objectifs :

* Identifier à partir d'une application industrielle un PIC.
* Elaborer un programme spécifique à une application à base de PIC

I-Mise en situation :

1-Fonctionnement du système : Porte automatique



L'ouverture et la fermeture automatique de la porte d'un garage d'automobile sont assurées par un système technique comportant les éléments suivants :

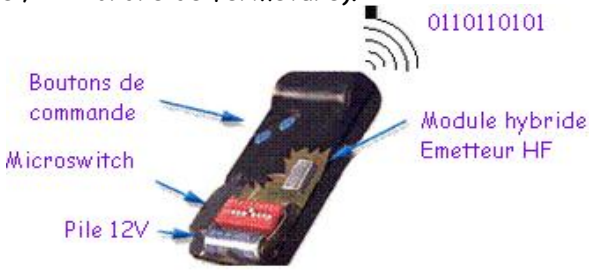
1. Télécommande à distance.
2. Moteur à courant continu + réducteur.
3. Carte de commande + pupitre.
4. Lampe de signalisation.
5. Antenne de réception.
6. Capteurs de fin de course.

Dans une première partie on va s'intéresser à la télécommande qui nous permet de commander la porte à distance, le principe est basé sur l'émission d'un code binaire sur dix bits [C9..C0] reconnaissable par la carte de commande de la porte. Ce code est composé de deux parties :

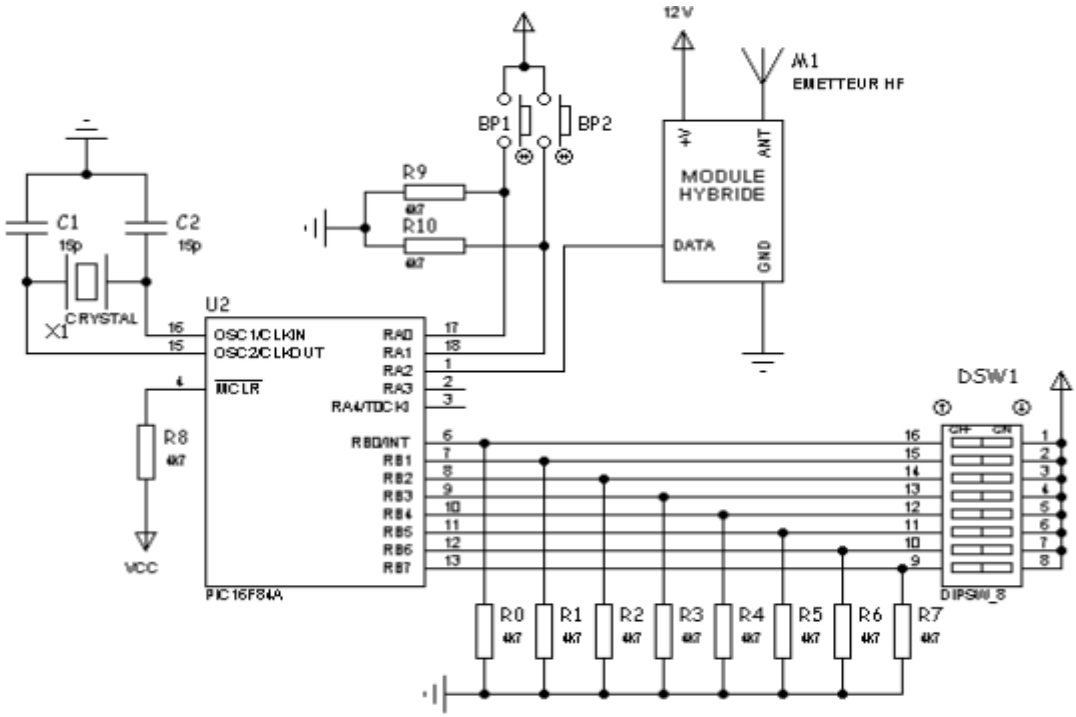
code d'identification de la télécommande								Ordre d'ouverture ~~/fermeture~~	
C9	C8	C7	C6	C5	C4	C3	C2	C1	C0
0	1	1	0	1	1	0	1		

code d'identification de la télécommande								Ordre ~~d'ouverture~~ /fermeture	
C9	C8	C7	C6	C5	C4	C3	C2	C1	C0
								1	1

Pour donner la possibilité à l'utilisateur d'adapter une nouvelle télécommande à la porte du garage ou de la remplacer en cas de panne, le constructeur a prévu un microswitch sur la télécommande qui nous permet de définir le code d'identification. Celui-ci est généralement inscrit sur la carte de commande ou donné dans le manuel d'utilisation de la porte. Le code correspondant à l'ordre d'ouverture ou de fermeture de la porte (C1, C0) est le suivant :
(**10** : Ordre d'ouverture ; **11** : ordre de fermeture).



www.BAC.org.tn



Le module hybride émetteur haute fréquence (M1) est un émetteur sans fil, qui à pour rôle de transmettre le code [C9...C0] généré par le microcontrôleur à la carte de commande de la porte. Ce module ne fera pas l'objet de notre étude.

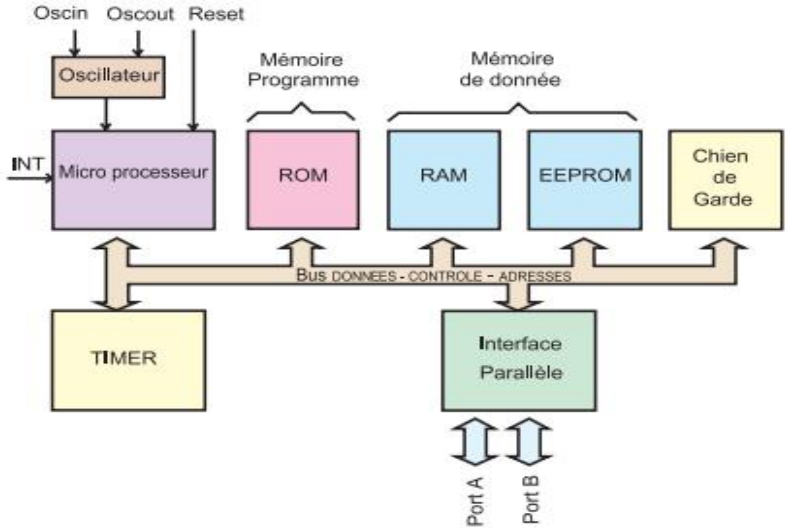
1-1 Activité 1 Page 74 :

1-2 Rappel sur les microcontrôleurs :

Un microcontrôleur se présente sous la forme d'un circuit intégré réunissant tous les éléments d'une structure à base de microprocesseur.

Voici généralement ce que l'on trouve à l'intérieur d'un tel composant :

1. Un microprocesseur (C.P.U.).
2. Une mémoire de donnée (RAM et EEPROM).
3. Une mémoire programme (ROM, OTPROM, UVPROM ou EEPROM).
4. Une interface parallèle pour la connexion des entrées / sorties.
5. Une interface série (synchrone ou asynchrone) pour le dialogue avec d'autres unités.
6. Des timers pour générer ou mesurer des signaux avec une grande précision temporelle.
7. Des convertisseurs analogique / numérique pour le traitement des signaux analogiques.



Structure interne PIC 16F84A

www.BAC.org.tn

Voir description de différent organe qui constitue le PIC page 101 manuel scolaire.

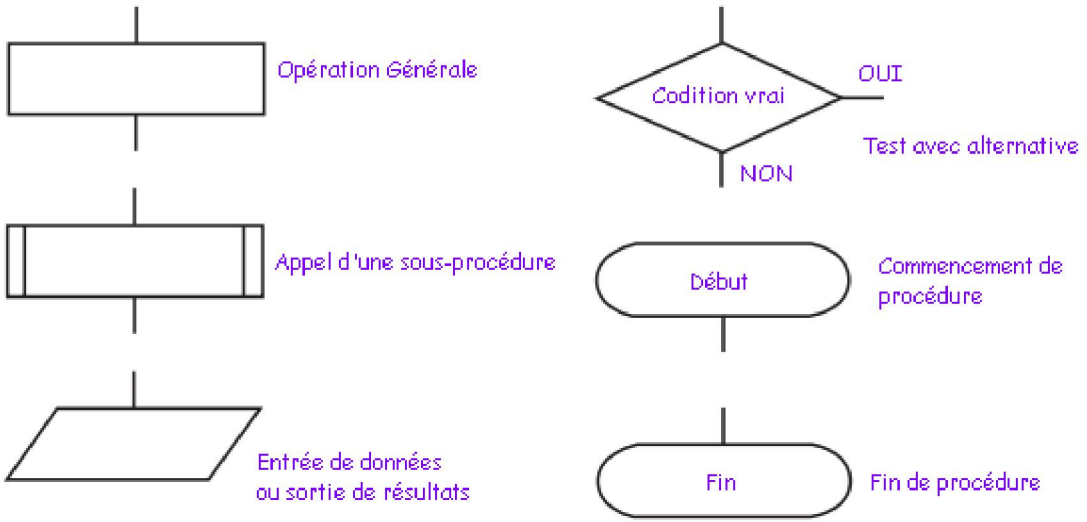
1-3 Activité 2 Page 75 :

II- Rappel sur la programmation graphique :

1- L'algorigramme ou l'organigramme :

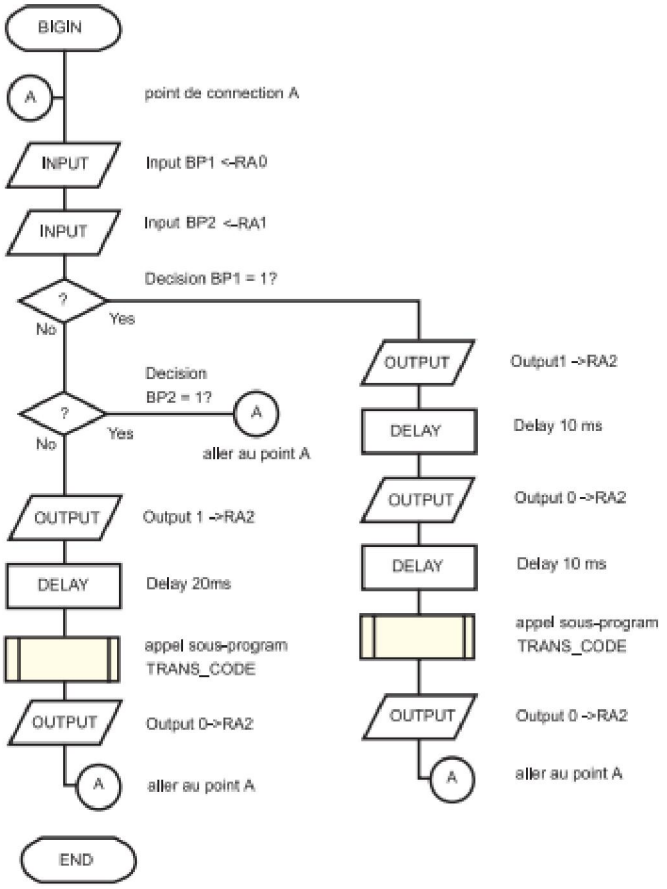
Définition : il s'agit d'une représentation graphique et normalisée utilisée pour analyser ou décoder un problème de logique.

Représentation normalisée: il s'agit de dessiner une suite de symboles définie comme suit :

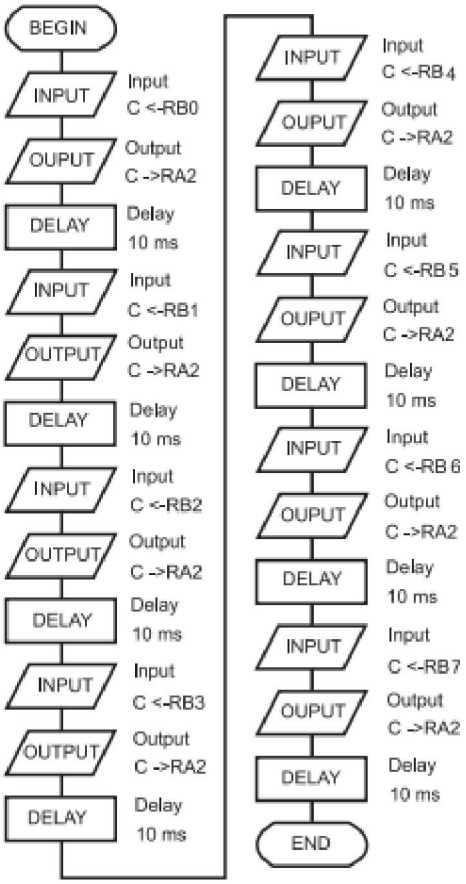


2-Application relative à la télécommande de la porte automatique :

Voici le programme du microcontrôleur de la télécommande réalisé par un logiciel graphique :



Programme principal



Sous Programme TRANS_CODE

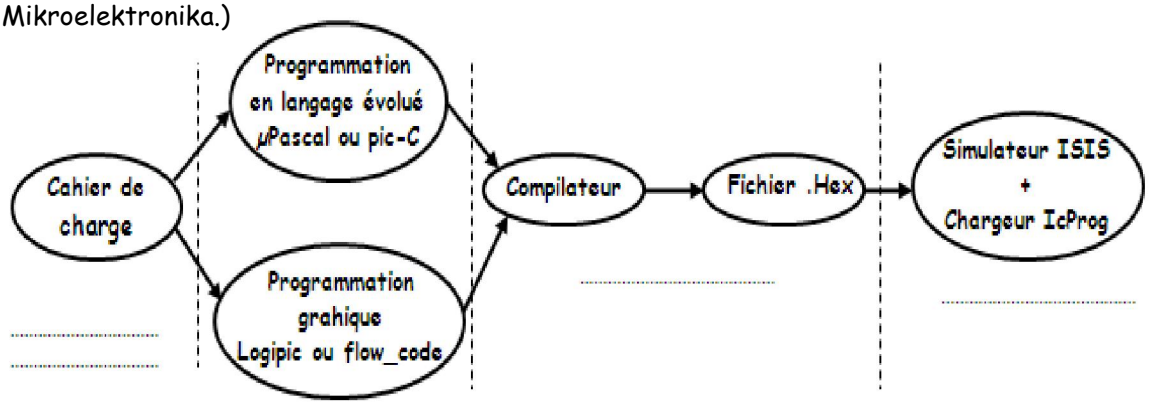
www.BAC.org.tn

2-1 Activité 3 Page 102 :

III-La programmation en langage évolué :

1-Méthode à suivre :

Que ce soit par la méthode graphique ou en langage évolué, l'écriture du programme ainsi que sa mise au point doivent suivre le diagramme suivant :
Il faut traduire le cahier des charges en une suite ordonnée d'actions que doit réaliser le processus de commande, cette suite d'opérations sera décomposée en actions élémentaires ou instructions c'est l'Algorithme. Par la suite il suffit de transformer cet algorithme en un langage évolué tel que le langage PASCAL ou le langage C.
Dans la suite du cours on s'intéressera au langage PASCAL. (Compilateur Mikropascal de Mikroelektronika.)



2- Activité 4 Page 78 et 79 :

3-Structure d'un programme :

Un programme est un texte que le compilateur va traduire en fichier hexadécimal. Alors il doit avoir une structure particulière. Le texte d'un programme contient au moins trois parties.

***L'entête :**

Ne contient qu'une ligne; commence par le mot réservé " Program " suivi du nom du programme.

***Les déclarations :**

Elles permettent de définir les éléments utilisés dans le programme. En effet on devra déclarer les variables utilisées pour permettre au compilateur d'effectuer les réservations de mémoire ainsi que les sous programmes (Procédures et fonctions).

***Le corps du programme :**

Commence par le mot réservé " Begin " et se termine par le mot réservé "End " suivi d'un point final. Ce qui suit ce "End" n'est pas pris en compte par le compilateur.
Entre "Begin" et "End" se trouvent les instructions à effectuer par le programme.

Algorithmique	Langage PASCAL	
Algorithme NomAlgorithme ;	Program NomProgramme ;	Entête
Variables	Var	// déclaration
Nomvariable Type	Nomvariable Type ;	
Constantes	Const	
Nomconstante Type = valeur	Nomconstante Type = valeur ;	
début	Begin	// Programme
.....	.....	// principal
.....	.....	
fin	End	

www.BAC.org.tn

3-1 Les Règles de bases :

*Toutes instructions ou actions se terminent par un point virgule ;

*Une ligne de commentaires doit commencer par "{" et se terminer par "}" ou commence par "//".

*Un bloc d'instructions commence par "Begin" et se termine par "End"

3-2 Les types de variables utilisées en Mikropascal :

Type	Désignation	Taille	Rang
octet	byte	8-bit	0 --->255
caractère	char	8-bit	0 --->255
Mot	word	16-bit	0 --->65535
Octet signé	short	8-bit	-128--->127
Entier	integer	16-bit	-32768--->32767
Entier long	longint	32-bit	-2147483648---> 2147483647
Réel	real	32-bit	$\pm$ 1.17549435082 * 10^{-38} .. $\pm$ 6.80564774407 * 10^{-38}
Tableau	Array[n] of type	n éléments	Rang du type
Chaîne de caractères	string[n]	n caractères	0 --->255

3-3 Les bases du compilateur Mikropascal :

Le décimal : A=10 ;

L'hexadécimal : A=$0F ; ou A=0x0F ;

Le binaire : A=%11010100 ;

3-4 Les opérateurs arithmétiques et logiques :

Opérateurs arithmétiques		Opérateurs de comparaison		Opérateurs de comparaison	
Opérateur	Opération	Opérateur	Opération	Opérateur	Opération
+	Addition	=	Egalité	AND	ET
-	Soustraction	<>	Différent	OR	OU
*	Multiplication	>	Supérieur	XOR	OU exclusif
/	Division	<	Inférieur	NOT	NON
div	Division entière	<=	Inférieur ou égale	SHL	Décalage à gauche
mod	Reste de la division entière	>=	Supérieur ou égale	SHR	Décalage à droite

3-5 Les structures usuels :

a-L'affectation : c'est l'action d'attribuer une valeur à une variable.

Algorithmique	Langage PASCAL
a←b+c	a:=b+c

www.BAC.org.tn

b-Les structures alternatives :

Algorithmique	Langage PASCAL

SI condition ALORS
DEBUT
Traitement ;
.....
FINSI ;
	IF condition THEN
	BEGIN
	Traitement ;
	END ;

SI condition ALORS
DEBUT
Traitement 1;
..... ;
FIN ;
SINON
DEBUT
Traitement 2;
..... ;
FINSI ;
	IF condition THEN
	BEGIN
	Traitement 1;
	..... ;
	END ;
	ELSE
	BEGIN
	Traitement 2;
	..... ;
	END ;

c-Les structures itératives ou répétitives :

Algorithmique	Langage PASCAL

I: entier;
.....;
POUR I <variant de valeur initiale>
JUSQU'A <valeur finale>
FAIRE
DEBUT
Traitement ;
.....
FINFAIRE ;
	I: integer;
	.....;
	FORI:=<valeur initiale> To <Valeur finale>
	Do
	BEGIN
	Traitement ;
	.....
	END ;

TANQUE condition FAIRE
DEBUT
Traitement ;
.....
FINFAIRE;
	WHILE condition DO
	BEGIN
	Traitement ;
	.....
	END;

d-Activité 5 Page 80-85 :

3-6 Les procédures et les fonctions :

Une suite d'instructions peut être rassemblée en un bloc qui peut être appelé depuis plusieurs endroits d'un programme. Ceci donne lieu aux notions de sous programme appelé aussi procédures ou fonctions.

***Procédures :**

Ce sont des groupes d'instructions qui vont former une nouvelle instruction simple utilisable dans un programme. En Pascal il faut les définir avant de les utiliser. Ceci se fait en utilisant une structure similaire à celle d'un programme.

Entête : Procédure Identificateur (Param1:Type1, Param2:Type2,...);

Déclarations : Déclarations de constantes, types, variables utilisés à l'intérieur de la procédure

Corps de la procédure : Begin
Instruction1; Instruction2;.....
End;

www.BAC.org.tn

***Fonctions :**

Une fonction est une procédure qui devra fournir un résultat de type numérique ou chaîne de caractères. La définition se fait en utilisant une structure similaire à celle de la procédure.

Entête : Function Identificateur (Param1:Type1, Param2:Type2,...):Type_R;

Déclarations : Déclarations de constantes, types, variables utilisés à l'intérieur de la fonction.

Corps de la fonction : Begin
Instruction1; Instruction2;.....
Identificateur:=résultat;
End;

3-7 Les fonctions adaptées aux microcontrôleurs PIC :

Le compilateur Mikropascal apporte une large bibliothèque de procédures et fonctions adaptées aux microcontrôleurs de la famille PIC de MICROCHIP. Ces fonctions sont accessibles dans l'aide du logiciel néanmoins on va citer quelques-unes.

Fonctions / Procédures	Exemple
Setbit (port, bit) ;	Setbit (portB, 2) ; mettre la broche RB2 à 1
Clearbit (port, bit)	Clearbit (portB, 5) ; mettre la broche RB5 à 0
Testbit (port, bit)	A:=testbit (portB, 7) ; affecter à la variable A l'état de RB7
Delay_ms (temps)	Delay_ms (150) ; attente de 150 ms
Button (port, bit, temps d'appui, état logique actif).	If Button(portA,2,10,1) then <Action 1> ; On teste l'appui sur un bouton poussoir relié à la broche RA2 pendant 10ms pour faire l'Action 1.

3-8 Structure générale d'un programme Mikro_Pascal

Program nom_programme ;
→Déclarations de constantes, types, variables utilisés dans le programme.
→ Déclarations des procédures et fonctions utilisées dans le programme.
Begin
→ Activation des interruptions utilisées
→Configuration des entrées/sorties.
→Initialisation des sorties utilisées.
while (1=1) do
begin
Instruction1;
Instruction2;
.....
end ;
end.
→ Déclaration des variables et des constantes :
Var nom_variable : type ;
Const nom_constante : type = valeur ;
→ Déclaration d'une procédure:
Procédure Identificateur (Param1:Type1, Param2:Type2,...);
Déclarations de constantes, types, variables utilisés à l'intérieur de la procédure
Begin
Instruction1;
Instruction2;.....
End;

www.BAC.org.tn

→ Déclaration d'une fonction :

Function Identificateur (Param1:Type1, Param2:Type2,...):Type_R;

Déclarations de constantes, types, variables utilisés à l'intérieur de la fonction.

Begin

Instruction1;

Instruction2;……..

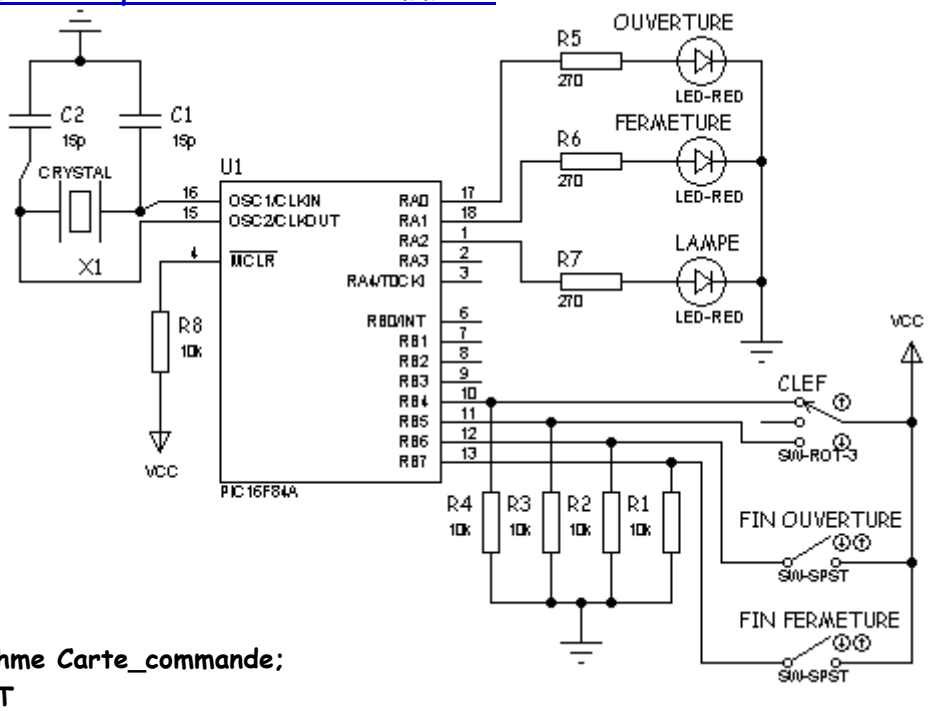
Identificateur:=résultat;

End;

IV-Applications :

a-Exemple1-Activité 6 Page 86-89 : Carte réel voir cous page 108

***Schéma électrique de la carte de commande :**



Algorithme Carte_commande;

_DEBUT

| TRISB ⇐ $……; // RA0, RA1, RA2 : sorties

| TRISB ⇐ $……; // tout le port b est configuré en entrée

| porta ⇐ 0; // initialisation des sorties

| **TANQUE (1=1) FAIRE** // boucle infinie

| _ **DEBUT**

| | **SI** (portB.4=……) **ALORS** // commutateur en position ouverture de la porte

| | **_DEBUT**

| | | **TANQUE** ((portB.6=0) ET (portB.4=1))

| | | **FAIRE** // tant que ordre d'ouverture de la porte

| | | // et capteur de fin de course non actionné

| | | **_DEBUT**

| | | | portA.0⇐……; // actionner moteur en rotation en sens d'ouverture

| | | | portA.2⇐……; // allumé la lampe de signalisation

| | | |**_FINFAIRE ;**

| | |**_FINSI ;**

| | **SI** (portB.5=1) **ALORS** // commutateur en position fermeture de la porte

| | _DEBUT

| | | **TANQUE** ((portB.7=……) ET (portB.5=……))

| | | **FAIRE** // tant que ordre de fermeture de la porte

| | | //et capteur de fin de course non actionné

www.BAC.org.tn

```
| | | _ DEBUT
| | | | portA.1⇐1;                    // actionner moteur en rotation en sens de fermeture
| | | | portA.2⇐1;                    // allumé la lampe de signalisation
| | | |_FINFAIRE;
| | |_FINSI ;
| | portA⇐0;                          // arrêter le moteur et éteindre la lampe
| |_FINFAIRE;
|_FIN.
```

***Saisir alors le programme en langage Mikropascal et tester son fonctionnement**

```
Program Carte_Commande;
Begin
| ……                                  // RA0, RA1, RA2 : sorties
| ……                                  // tout le port b est configuré en entrée
| ……                                  // initialisation des sorties
| ……                                  // boucle infinie
| _ ……
| |……                                 // commutateur en position ouverture de la porte
| | _……
| | | ……
| | | ……                              // tant que ordre d'ouverture de la porte
| | |                                  // et capteur de fin de course non actionné
| | | _……
| | | | ……                           // actionner moteur en rotation en sens d'ouverture
| | | | ……                           // allumé la lampe de signalisation
| | | |_……
| | |_……
| | ……                                // commutateur en position fermeture de la porte
| | _……
| | | ……
| | |……                               // tant que ordre de fermeture de la porte
| | |                                  //et capteur de fin de course non actionné
| | | _ ……
| | | | ……                           // actionner moteur en rotation en sens de fermeture
| | | | ……                           // allumé la lampe de signalisation
| | | |_……
| | |_……
| | ……                                // arrêter le moteur et éteindre la lampe
| |_……
|_End.
```

b-Exemple2 : Commande du monte-charge

L'appui sur le bouton départ cycle **dcy** provoque :

*****Montée de la cabine vers le 2ème étage par la rotation du moteur dans le sens 1 **(CM)** jusqu'à l'action sur le capteur **m**.

*****Une attente de **T=5s**.

*****Descente de la cabine au rez de chaussée par la rotation du moteur dans le sens 2 **(CD)** jusqu'à l'action sur le capteur **d**.

On donne :

www.BAC.org.tn

☑ GRAFCET de point de vue PC :

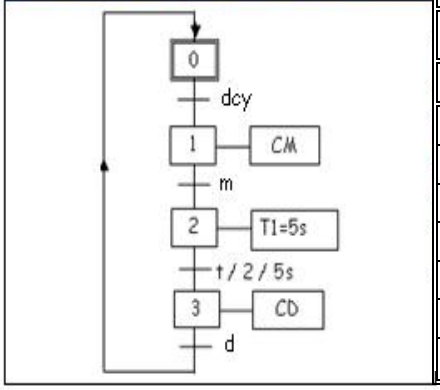
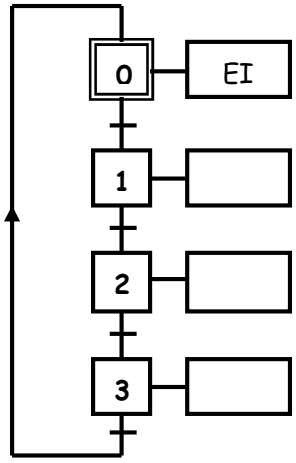


tableau d'affectation	
Entrées/Sorties	**Broche µc**
Entrée: **dcy**	RA0
Entrée: **m**	RA1
Entrée: **d**	RA2
Sortie: **EI**	RB0
Sortie: **CM**	RB1
Sortie: **T1**	RB2
Sortie: **CD**	RB3



GRAFCET codé µC

*** Terminer la saisie du programme monte charge et créer le fichier d'extension<.Hex>.**
*** Tester à l'aide d'un logiciel CAO (Proteus ISIS) son fonctionnement.**
*** Transférer le fichier d'extension<.Hex> vers le PIC 16F84A en utilisant les kits, logiciel et matériel adéquats.**

Algorithme Monte_Charge	Commentaires	program Monte_Charge;
variables X0, X1, X2, X3, t1:……;		var X0, X1, X2, X3, t1: byte;
début		begin
TrisA<---……;	Configuration des broches RA0, RA1, RA2 en ……	TrisA:=$……;
……<---……;	Configuration des broches RB0, RB1, RB2, RB3 en sorties	TrisB:=$……;
PortB<---……;	…… du PortB	PortB:=0;
X0<---……;		X0:=1;
X1<---……;		X1:=0;
……<---……;		X2:=……;
X3<--- 0;		X3:=……;
……<---……;		t1:=……;
Tant que (…=…) faire		while true do
début		begin
Si ((PortA.0=1)…… (……) alors		if ((PortA.0=……) and (X0=1))then
Début		begin
X0<---……;	Désactivation X0	X0:=……;
X1<---……;	Activation X1	X1:=1;
Fin si;		end;
……		if ((PortA.1=1) and (X1=1))then
Début		begin
X1<---……;		X1:=0;
X2<---……;	Activation X2	X2:=……;
Fin si;		end;

www.BAC.org.tn

		if ((X2=1) and (t1=……))then
		begin
		X2:=0;
		X3:=1;
		end;
		if ((PortA.2=1) and (X3=1))then
		begin
		X3:=0;
		X0:=1;
		end;
		if (X0=1) then PortB.0:=1 else PortB.0:=0;
		if (X1=1) then PortB.1:=1 else PortB.1:=0;
		if (X2=1) then
		begin
		PortB.2:=……;
		t1:=0;
		delay_ms (……);
		t1:=……;
		end else
		begin
		PortB.2:=……;
		t1:=0;
		end;
		if (X3=……) then PortB.3:=1 else PortB.3:=0;
		end;
		end.

www.BAC.org.tn